## Московский государственный технический университет им. Н.Э. Баумана.

Факультет «Системы обработки информации и управления» Кафедра ИУ5

Отчёт

## по лабораторной работе №5

по дисциплине «Разработка интернет приложений»

«**Работа с СУБД**»

Студент: Акушко А.С. Группа РТ5-51

Преподаватель: Гапанюк Ю.Е.

Москва, 2019

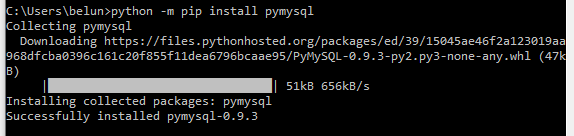
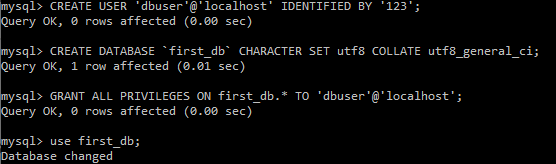
**Задание и порядок выполнения ЛР №5**

В этой лабораторной работе вы познакомитесь с популярной СУБД MySQL, создадите свою базу данных. Также вам нужно будет дополнить свои классы предметной области, связав их с созданной базой. После этого вы создадите свои модели с помощью Django ORM, отобразите объекты из БД с помощью этих моделей и ClassBasedViews.

Для сдачи вы должны иметь:

1. Скрипт с подключением к БД и несколькими запросами.
2. Набор классов вашей предметной области с привязкой к СУБД (класс должен уметь хотя бы получать нужные записи из БД и преобразовывать их в объекты этого класса)
3. Модели вашей предметной области
4. View для отображения списка ваших сущностей

**Работа с MySQL:**



**Код программы**

**connection.py**

**import** pymysql pymysql.install\_as\_MySQLdb()

**class** Connection:

**def** init (self, user, password, db, host=**'localhost'**): self.host = host

self.user = user self.password = password self.db = db self.charset = **"utf8"** self.\_connection = None

@property

**def** connection(self):

**return** self.\_connection

**def** enter (self): self.connect()

**def** exit (self, exc\_type, exc\_val, exc\_tb): self.disconnect()

**def** connect(self):

**if not** self.\_connection: self.\_connection = pymysql.connect(

host=self.host, user=self.user, password=self.password, db=self.db, charset=self.charset

)

**def** disconnect(self):

**if** self.\_connection: self.\_connection.close()

**class** Product:

**def** init (self, db\_connection, name, description): self.db\_connection = db\_connection.connection self.name = name

self.description = description

**def** save(self):

c = self.db\_connection.cursor()

c.execute(**"INSERT INTO 'products' ('name', 'description') VALUES (%s, %s);"**, (self.name, self.description))

self.db\_connection.commit() c.close()

connection = Connection(**'dbuser'**, **'123'**, **'first\_db'**, **'localhost'**) **with** connection:

product = Product(connection, **'Computer'**, **'Electronic desktop device for working with files and programming'**)

product.save()

**dbshow.py**

**import** pymysql pymysql.install\_as\_MySQLdb()

db = pymysql.connect ( host=**"localhost"**, user=**"dbuser"**, passwd=**"123"**, db=**"first\_db"**

)

*# курсор*

c = db.cursor()

c.execute(**'SELECT** *\** **FROM products;'**) entries = c.fetchall()

**for** e **in** entries: print(e)

c.close() db.close()

### views.py

**from** django.shortcuts **import** render

**from** django.http **import** HttpResponse

**from** django.views.generic **import** View, ListView

**from** lab5.models **import** Product, Review

**import** math

*# Список продуктов*

**class** ListProductView(ListView): model = Product

template\_name = **'product\_list.html'** context\_object\_name = **'products'** paginate\_by = 3

**def** get(self, request, page=1):

*# Количество продуктов на странице*

elements\_on\_page = 9

*# Количество продуктов в строке*

elements\_in\_row = 3

products = Product.objects.all()

pages\_count = math.ceil(len(products) / elements\_on\_page)

start\_index = (int(page) - 1) \* elements\_on\_page end\_index = start\_index + elements\_on\_page products = products[start\_index:end\_index]

index = 1 rows = [] row = []

**for** product **in** products: row.append(product)

**if** index == elements\_in\_row: rows.append(row)

row = [] index = 1

#### else:

index += 1

**if** len(row) > 0: rows.append(row)

**return** render(request, **'product\_list.html'**, {**"products"**: rows, **"page"**: page, **"pages\_count"**: pages\_count})

*# Страница с информацией о продукте и отзывами*

**class** ProductView(View):

**def** get(self, request, product\_id): elements\_in\_row = 2

product = Product.objects.get(id=product\_id)

reviews = Review.objects.filter(product\_id=product\_id) reviews\_count = len(reviews)

index = 1 rows = [] row = []

**for** review **in** reviews: row.append(review)

**if** index == elements\_in\_row: rows.append(row)

row = [] index = 1

#### else:

index += 1

**if** len(row) > 0: rows.append(row)

**if** len(rows) == 0: rows = **None**

**return** render(request, **'product.html'**, {**"product"**: product, **"reviews"**: rows, **"reviews\_count"**: reviews\_count})

### models.py

*# coding=utf-8*

**from** django.db **import** models

**from** django.contrib.auth.models **import** User

**class** Product(models.Model):

*# Название товара*

name = models.CharField(max\_length=255)

*# Описание товара*

description = models.CharField(max\_length=1000)

*# Продавец*

seller = models.CharField(max\_length=255)

*# Короткое описание товара*

**def** short\_description(self):

**return** self.description[:126]

**def** str (self): **return ' '**.join([

self.name,

**' from '**, self.seller,

])

**class** Review(models.Model):

*# Пользователь, который оставил отзыв*

user = models.ForeignKey(User, on\_delete=models.CASCADE)

*# Товар, под которым оставлен отзыв*

product = models.ForeignKey(Product, on\_delete=models.CASCADE)

*# Текст отзыва*

description = models.CharField( max\_length=500,

)

**def** str (self): **return ' '**.join([

**'review \''**, str(self.description), **' \' from user @'**,

str(self.user.username),

])

### settings.py

DATABASES = {

**'default'**: {

#### 'ENGINE': 'django.db.backends.mysql', 'NAME': 'first\_db',

**'USER'**: **'dbuser'**,

**'PASSWORD'**: **'123'**,

**'HOST'**: **'localhost'**, **'PORT'**: 3306,

**'OPTIONS'**: {**'charset'**: **'utf8'**}, **'TEST\_CHARSET'**: **'utf8'**,

}

### urls.py

**from** django.conf.urls **import** url

**from** django.contrib **import** admin

**from** lab5.views **import** ProductView, ListProductView urlpatterns = [

url(**r'^admin/'**, admin.site.urls),

url(**r'^product/(?P<product\_id>\d+)'**, ProductView.as\_view()), url(**r'^$'**, ListProductView.as\_view(), name=**'base\_template'**),

]

# Скриншоты выполнения

